PHP string functions as template operators

Source file: doc/eng-us/features/6.0/string-template-operators.md Last changed 2026-11-04
https://doc.exponential.earth/learninminutes/exphelp/viewer/features/6.0/string-template-operators.md

This page is for template developers who need to work with text in a template. Sixty PHP string functions (62 on PHP 8.3 or newer, see below) are available in templates as operators, so a template can measure, search, cut, compare and format text without a custom extension. Added in March 2026 (release 6.0.13) in the class `eZTemplateStringsOperator` (`lib/eztemplate/classes/eztemplatestringsoperator.php`), with a PHPUnit suite of about a thousand lines (`tests/tests/lib/eztemplate/eZTemplateStringsOperatorTest.php`, see PHPUnit 13).

How an operator maps to PHP

The piped value is the first PHP argument (the "subject"). The template arguments are the remaining PHP arguments, in the order PHP defines them. `{$text|substr(0, 5)}` is `substr( $text, 0, 5 )`.

Template

```
{def $text = 'Hello, Exponential'}
{$text|strlen}                       {* 18 *}
{$text|strpos('Exp')}                {* 7 *}
{$text|substr(7)}                    {* Exponential *}
{$text|substr(0, 5)}                 {* Hello *}
{$text|str_contains('Expo')}         {* true *}
{$text|str_starts_with('Hello')}     {* true *}
{$text|lcfirst}                      {* hello, Exponential *}
{1234567.891|number_format(2, '.', ',')}   {* 1,234,567.89 *}
{'%s has %d items'|sprintf($name, $count)}
{'<b>bold</b> text'|strip_tags}      {* bold text *}
```

`strpos`, `stripos`, `strrpos` and `strripos` return `false` when nothing is found; test with `eq( $pos, false() )` and not `eq( $pos, 0 )`. A missing required argument produces a template warning that names the argument (for example "Missing required parameter: start") instead of a PHP fatal error.

The two replace operators: read this once

```
{'Red red RED'|ristring('red', 'blue')}   {* Red blue RED   case-sensitive *}
{'Red red RED'|rstring('red', 'blue')}    {* blue blue blue case-insensitive *}
```

`ristring` is PHP `str_replace` (case-sensitive) and `rstring` is `str_ireplace` (case-insensitive). The names are easy to mix up; both accept arrays of search and replacement strings. A null subject (an unset attribute) becomes an empty string in `ristring`.

The full list

Without parameters (the compiler inlines constants and emits a plain PHP call): `addslashes`, `bin2hex`, `convert_uudecode`, `convert_uuencode`, `hex2bin`, `lcfirst`, `quoted_printable_decode`, `quoted_printable_encode`, `quotemeta`, `soundex`, `str_shuffle`, `stripcslashes`, `strlen`, `stripslashes`.

With parameters: `addcslashes`, `chunk_split`, `hebrev`, `html_entity_decode`, `htmlentities`, `htmlspecialchars_decode`, `levenshtein`, `ltrim`, `metaphone`, `number_format`, `rtrim`, `similar_text`, `sprintf`, `str_contains`, `str_ends_with`, `str_getcsv`, `str_split`, `str_starts_with`, `str_word_count`, `strcasecmp`, `strcmp`, `strcoll`, `strcspn`, `strip_tags`, `stripos`, `stristr`, `strnatcasecmp`, `strnatcmp`, `strncasecmp`, `strncmp`, `strpbrk`, `strpos`, `strrchr`, `strripos`, `strrpos`, `strspn`, `strstr`, `strtok`, `strtr`, `substr`, `substr_compare`, `substr_count`, `substr_replace`, `vsprintf`.

Plus the creative additions `ristring` and `rstring`, and, registered only when PHP provides the functions (8.3 or newer), `str_increment` and `str_decrement` (no parameters). Check the list your installation really has with the expinfo operator.

`number_format` follows PHP: with one parameter it gives the decimals; with three, decimals, decimal point and thousands separator (PHP needs the last two together). `sprintf` uses the piped string as the format and passes the template arguments to it.

What is deliberately not here

Operators that already existed keep their old behaviour and are not redefined: `chr`, `ord`, `trim`, `nl2br`, `rot13`, `crc32`, `md5`, `sha1`, `concat`, `indent`, `upcase`, `downcase`, `count_chars`, `count_words`, `break`, `wrap`, `shorten`, `pad`, `upfirst`, `upword`, `simplify`, `wash`, `append`, `prepend`, `merge`, `contains`, `compare`, `extract`, `extract_left`, `extract_right`, `begins_with`, `ends_with`, `implode`, `explode`, `repeat`, `reverse`, `insert`, `remove`, `replace`, `unique`, `array_sum`. Prefer those for what they do; for example `{$text|upcase}` and not a PHP-flavoured spelling.

Before you upgrade

Operator names are global. If one of your extensions already defines an operator with one of the new names (`strpos`, `substr`, `strlen` and `sprintf` are the likely ones), test the pages that use it after the update: list the operators an installation provides with the expinfo operator and compare.

Related pages

- Role and policy operators
- The expinfo operator, PHPUnit 13
- Changelog 6.0.13
- Chronicle: March 2026

> **Become an Exponential Partner**
>
> Partners share their information, provide funding, contribute back and develop the products with us, and that is how Exponential grows. A partner website and dashboard are planned for mid-2027: please become a partner today to speed up our growth rate and

> land new features faster.
>
> Partners: start here | How to apply

Exponential Copyright © 1998-2026 Exponential Foundation and 7x
Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".
This documentation is free documentation, licensed under the GNU Free Documentation License 1.3 or later:
https://www.gnu.org/licenses/fdl-1.3.html.